Music Coach – An Adaptive Software Application for Piano Practice

Jemima Orakwue

Supervisor: Fiona Knight

The Problem

Traditional Route

- Private Lessons → Expensive
- Teachers determine specific student needs using pre-existing pedagogical knowledge
- Plenty of resources → Significant user willpower required

Gamified Apps

- Commercial apps function like binary score games, not interactive teachers
- Applications provide binary “hit or miss” scoring without diagnostic feedback on error causes

Project Aim

To develop a Music Coach that analyses a user's piano performance to provide adaptive feedback and personalised practice recommendations.

Objectives

- Analyse techniques for extracting performance features to determine the most accurate method for real-time error detection.
- Implement a full-stack application to include:
  - a feedback system capable of identifying pitch, timing, and dynamic inaccuracies.
  - a recommendation mechanism that suggests specific exercises, or scales based on a learner's historical performance patterns.
  - a responsive user interface that visualises errors, performance comparisons, and progress history in an accessible way.
  - a database to store user and music data.
- Conduct a quantitative user survey to validate the market need and identify the specific barriers faced by different types of learners.
- Evaluate the application through user testing to assess usability and the reliability of the detection system.

Research Methods

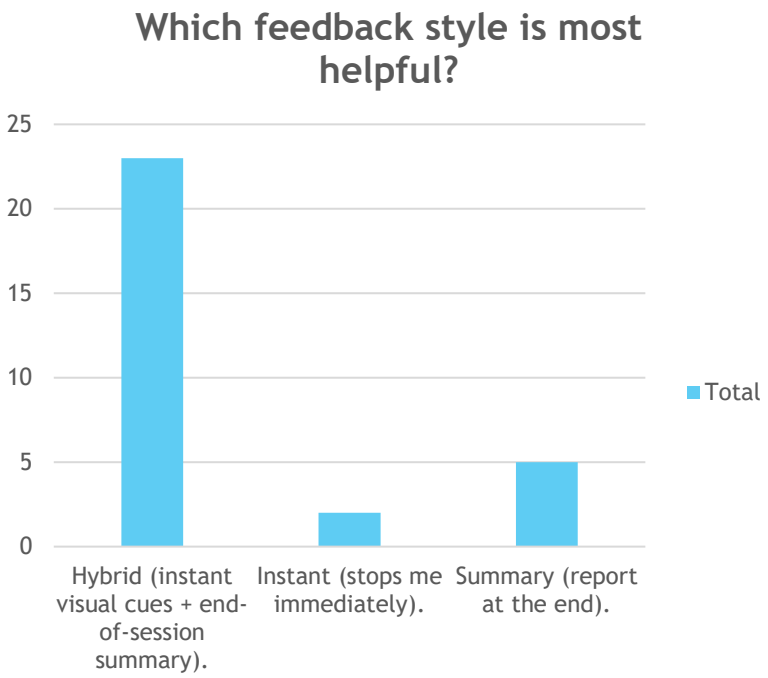
- Comparative Technical Analysis
  - Determine technical feasibility – audio vs Musical Instrument Digital Interface (MIDI)
- Comparative Market Analysis
  - Identify & validate the Pedagogical Gap
- Quantitative User Survey
  - Identify & validate the Pedagogical Gap
- Experimental Prototyping
  - Determine technical feasibility – audio vs Musical Instrument Digital Interface (MIDI)
- Usability Testing
  - User Interface Review
  - Formative and Summative Testing

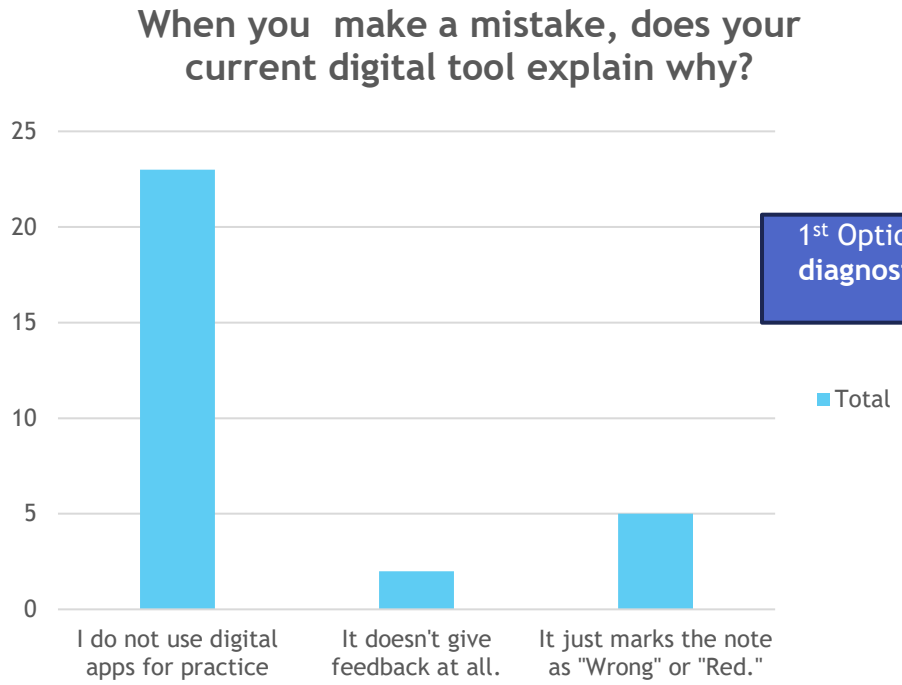
Market Analysis Results

Feature	Application/Website						
	Simply Piano	Yousician	Flowkey	Keysnake	Museflow	ArtMaster	Leopold AI
Chords detection	✓	✓	✓	×	✓	×	✓
Rhythm/ Timing Analysis	✓	✓	×	✓	✓	×	✓
MIDI Input Support	✓	✓	✓	✓	✓	×	×
Real-Time Error Flagging	✓	✓	✓	✓	✓	×	×
Wait Mode (Stops until correct note)	✓	✓	✓	✓	×	×	×
Diagnostic Feedback	×	×	×	×	×	×	! (Post)
Standard Sheet Music Display	✓	! (Scroll)	✓	×	✓	×	×
Generative/ Infinite Drills	×	×	×	×	✓	×	✓
Adaptive Difficulty	! (Basic)	! (Basic)	! (Partial)	✓	✓	×	✓
Curriculum Structure	Linear	Linear	Self-directed	Unstructured	Adaptive	Instructor-Led	Adaptive

Table 1 - Technical Feature Matrix identifying critical pedagogical gaps in current market solutions, (✓= Feature present, × = Feature absent, ! = Limited functionality)

Validation and Research

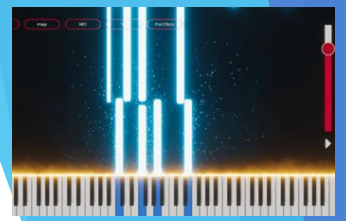




1st Option: **"It provides a diagnostic feedback"** - 0 selected

0% of users receive diagnostic feedback from current tools, validating the need for an Intelligent Tutoring System

Validation and Research Key Findings



Source:https://devforum.roblox.com/t/sheets-a-free-sheet-music-experience/3678800

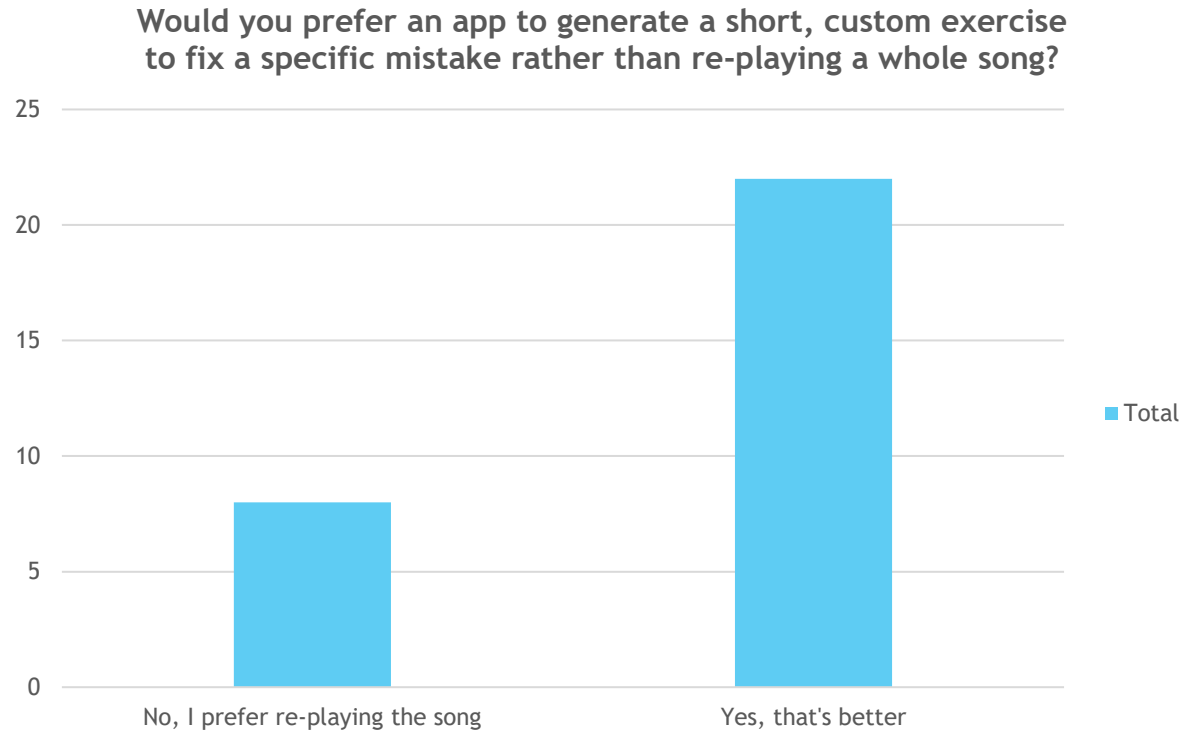
1. Pedagogical Gap

- Comparative analysis confirms that while high-quality static resources exist, commercial "gamified" apps prioritise binary scoring (hit/miss) over diagnostic feedback.
- Research identifies a critical need for a system that moves beyond simple scoring to explain *why* errors occur, providing a true pedagogical feedback loop.

2. Adaptive Framework Decision

- Intelligent Tutoring Framework (ITS): Following the Gomes (2024) model, the system is designed with a Student Model (to track progress) and a Pedagogical Module (to guide learning).
- Hybrid Feedback Strategy: Implements a multi-stage data flow:
  - Performance Capture → Student Model Data → Pedagogical Logic → UI Output.
- Hybrid Feedback System (Xu and Zulkarnain, 2025)

Justification For Generating Drills



Data Ingestion and Analysis Key Findings

3. Input Data Feasibility

- Audio Analysis introduces significant latency and struggles with polyphony applications (Schedl, et al, 2014).
- Utilising standard Musical Instrument Digital Interface (MIDI) protocols provides 100% accurate, low-latency symbolic data capture (Schedl, et al, 2014)
- The application architecture was designed strictly around real-time WEB MIDI stream data processing to capture exact pitch, velocity, and timestamps.

4. Evaluation Logic & Criteria

- Criteria for evaluating performance (Huang and Ding, 2022) :
  - Overall evaluation → success or failure based on the total error count.
  - Rhythm evaluation → accuracy of note duration against the standard score.
- Extracted performance data used for drill generation

Source : https://www.gear4music.com/

Design and Implementation of Artefact

Requirements - Functional Requirements

- Universal USB MIDI keyboard connectivity.
- Static musical notation tracking with live visual feedback.
- Persistent historical performance metrics and pedagogical summaries.
- Real time extraction of Note On/Off messages (Pitch, Velocity, Timestamps).
- Alignment algorithms to map live input against reference MIDI arrays.
- Error detection algorithm - separating pitch errors from timing discrepancies.
- Automated generation of targeted, short practice exercises.

Scoping And Compromises

- Missing and skipped notes detection
- Adaptive difficulty toggles (Beginner versus Advanced).
- Evaluate and test the rule-based AI using example scenarios and test cases.
- Implement testing for each major system component
- Categorise exercises based on the type of detected mistakes.

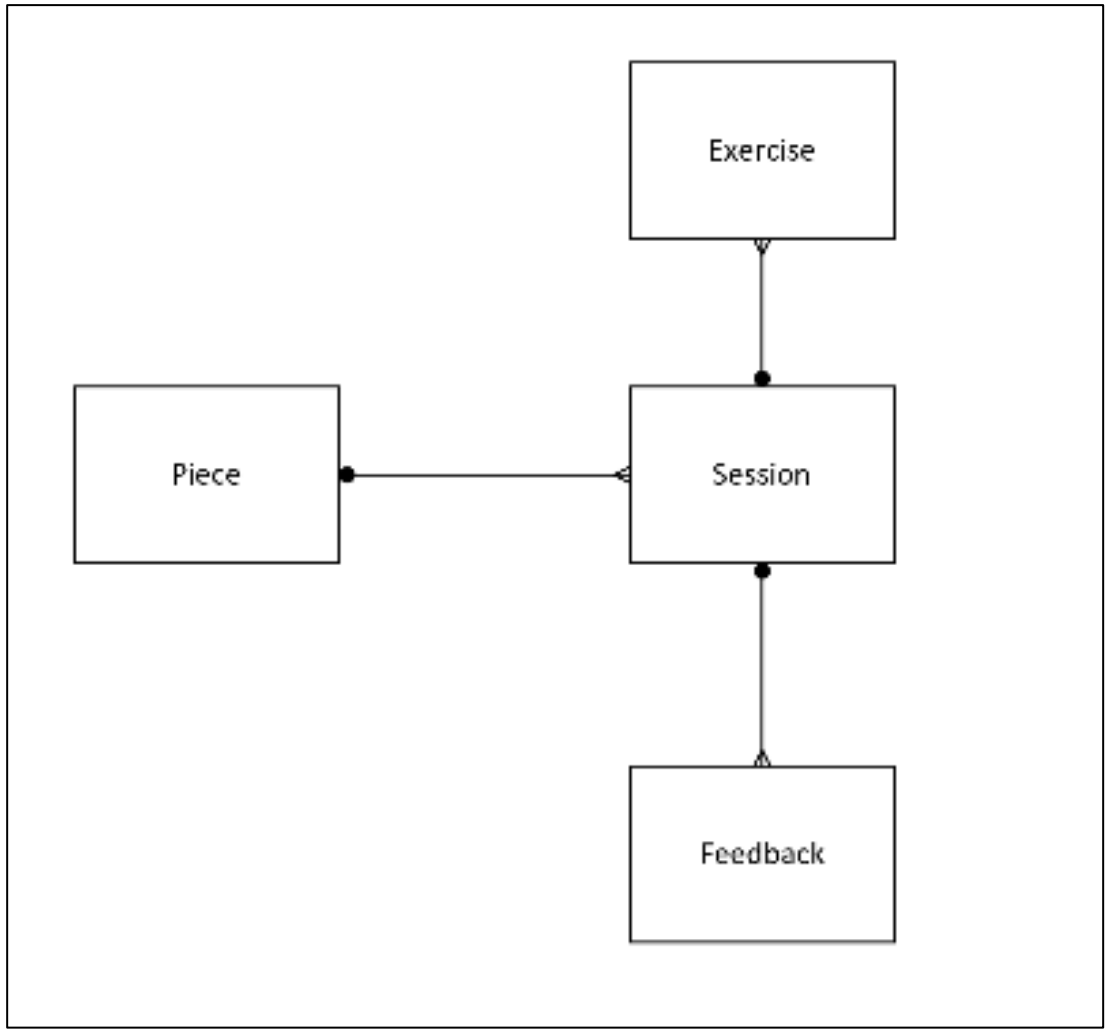
Requirements - Non-Functional

- System delay must be maintained below 25ms.
- 99% MIDI capture accuracy over a continuous practice session.
- The interface must follow a theory first design approach to avoid the visual dependency commonly associated with gamified music applications.
- Row Level Security (RLS) to ensure that practice history is accessible only to the authenticated user.

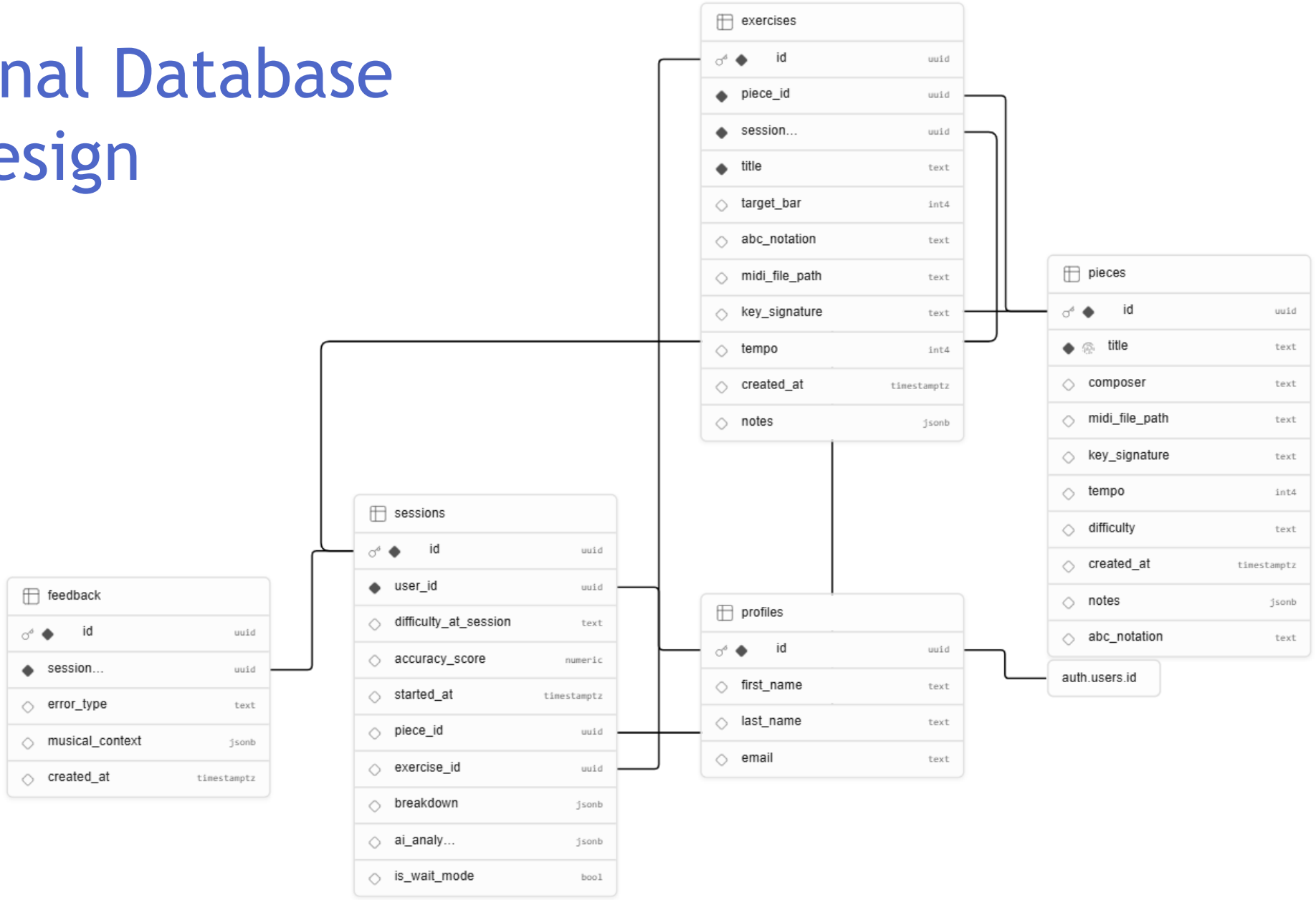
System Flow


Piano Input
Performance Analysis
Data Persistence
Pedagogical Feedback
UI

Database Design



Final Database Design



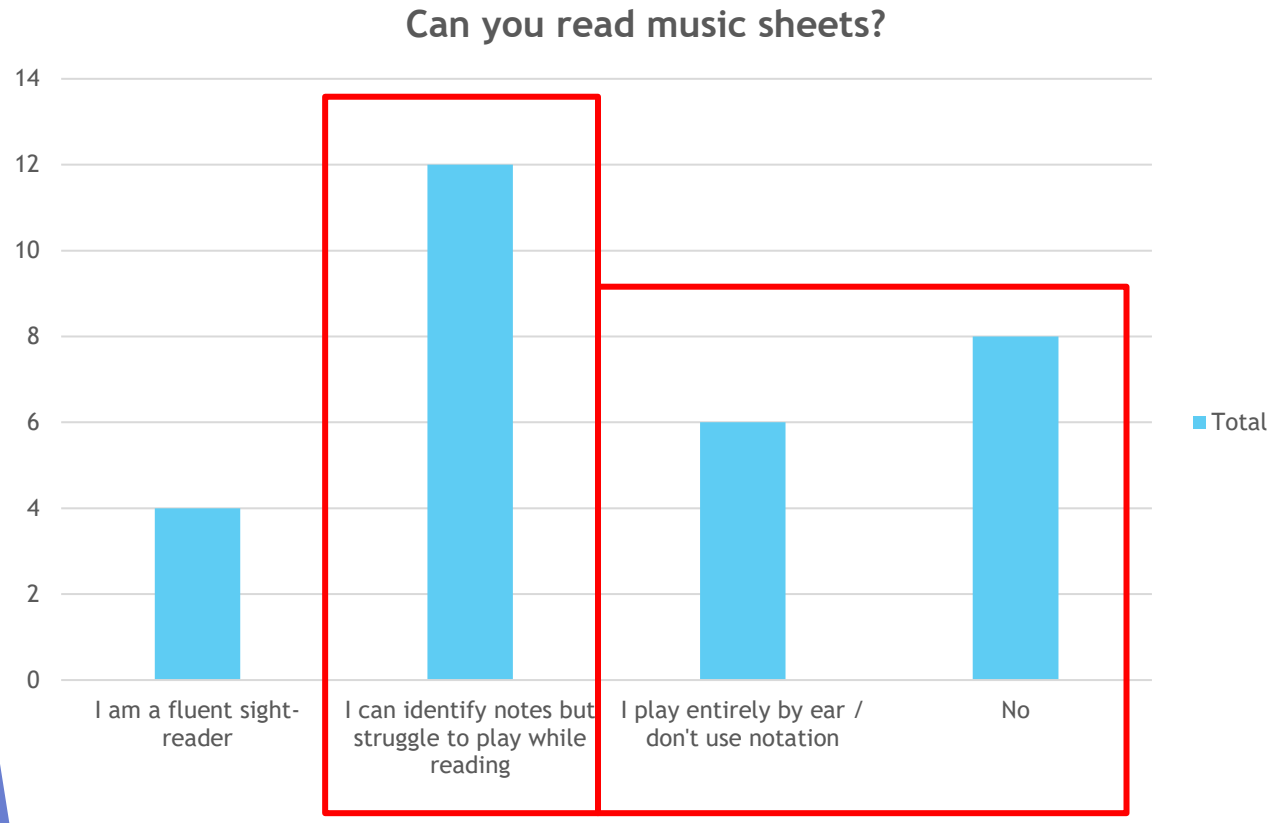
Technology stack

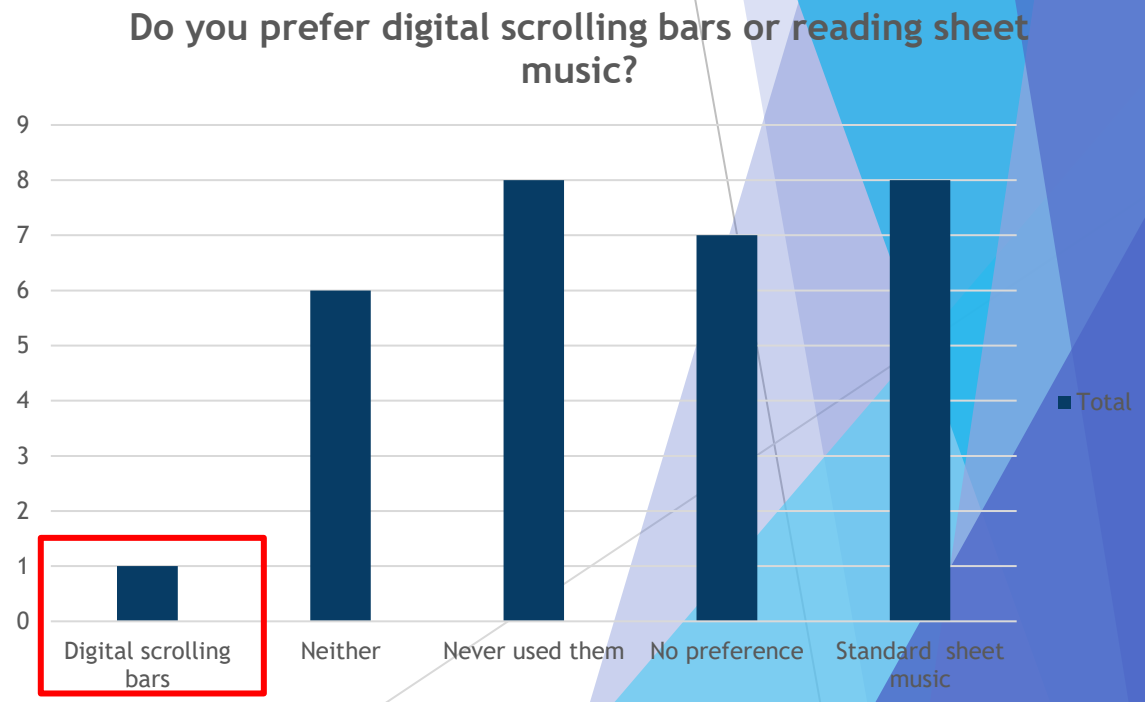
- **Next.js** utilised for high performance rendering and state management.
  - Python → TypeScipt
  - **Tonal.js, webmidi.js, abcjs, shadcn**
- **Supabase** serves as the knowledge base and student model.
  - PostgreSQL
  - Implementation of Row Level Security (RLS)
- **Google Gemini API** provides the pedagogical module for adaptive coaching
  - Utilised to translate raw symbolic error objects into structured, conte
    precise text advice

https://images.icon-icons.com/2389/PNG/512/next_js_logo_icon_145038.png
https://upload.wikimedia.org/wikipedia/commons/thumb/4/4c/Typescript_logo_2020.svg/3840px-Typescript_logo_2020.svg.png?utm_source=commons.wikimedia.org&utm_campaign=index&utm_content=thumbnail
https://www.supabase.com

Notation Decision

High user demand for traditional sheet music over scrolling bars





Comparison algorithm

- USB connection to parse real-time MIDI event messages.
  - Custom scripts extract pitch, velocity, and duration from raw MIDI buffer messages → Baseline
- Input MIDI notes are compared against a baseline.
- The system calculates the timing between the input note timestamp and the ideal temporal position in the reference file (reference note JSON).
- Records every error type and compiled into a persistent feedback state array
  - The system categorises whether the error was timing or pitch.

```
export interface ReferenceNote{
  midi_number: number;
  note_name: string;
  octave: number;
  onset_ticks: number;
  onset_seconds: number;
  duration_ticks: number;
  duration_seconds: number;
  velocity: number;
  bar: number;
  beat: number;
}
```

Feedback for each note per session

error_type	musical_context jsonb
Pitch	{"bar_index":5,"advice_text":"You played a A4 ins
Pitch	{"bar_index":2,"advice_text":"You played a C4 ins
Pitch	{"bar_index":1,"advice_text":"You played a E4 ins
Pitch	{"bar_index":1,"advice_text":"You played a F4 ins
Timing	{"bar_index":2,"advice_text":"You played this not
Timing	{"bar_index":2,"advice_text":"You played this not

The Drill Generation Algorithm

1. The comparison algorithm identifies the bar with the highest error density.
2. An algorithm extracts the specific bar data to create a targeted, short practice exercise.
3. The errors → AI prompting framework to provide contextually relevant remediation advice.

```
export function identifyWorstBar(errorLog: PracticeError[]): number | null {
  if (!errorLog || errorLog.length === 0) return null;

  const barMistakeCounts: Record<number, number> = {};

  errorLog.forEach(error => {
    if (error.bar) {
      barMistakeCounts[error.bar] = (barMistakeCounts[error.bar] || 0) + 1;
    }
  });

  let worstBar = null;
  let maxMistakes = 0;

  for (const [barStr, count] of Object.entries(barMistakeCounts)) {
    if (count > maxMistakes) {
      maxMistakes = count;
      worstBar = parseInt(barStr);
    }
  }
  return worstBar;
}
```

```
export async function POST(req: Request) {
  try {
    const { errors, pieceData, sessionId } = await req.json();

    if (!errors || !sessionId) {
      return NextResponse.json({ error: "Errors and sessionId are required." }, { status: 400 });
    }
```

AI Prompting

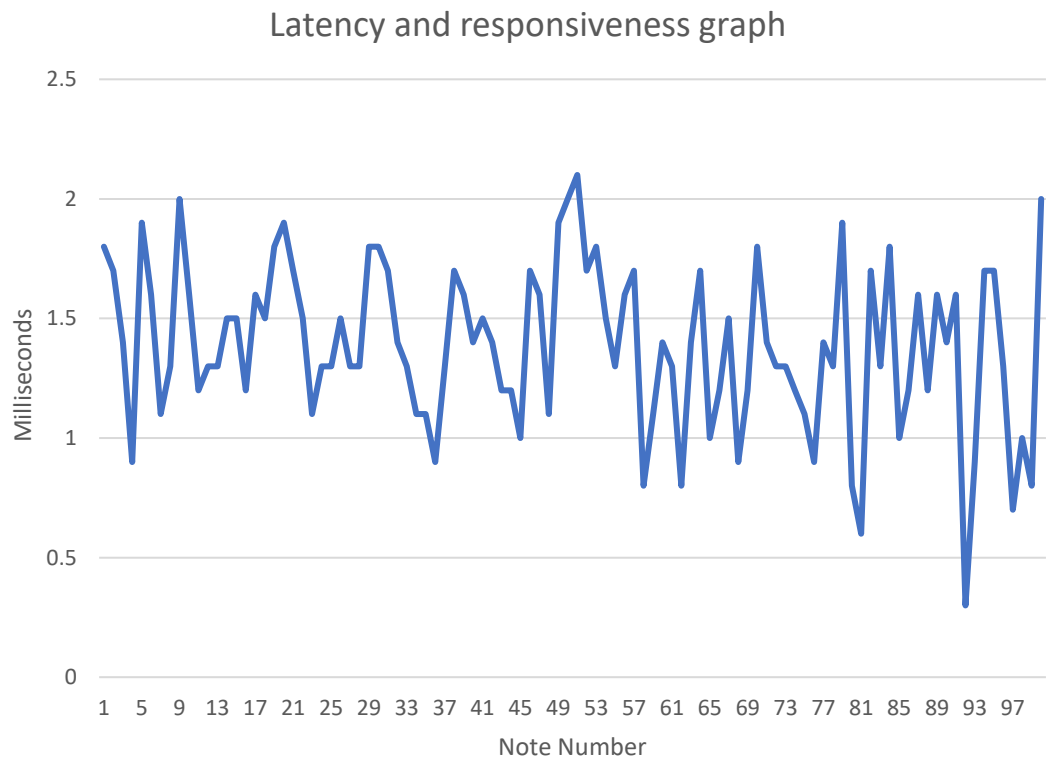
- The prompt constrains the AI to act as a supportive, technically precise piano tutor.
- The AI is fed feedback data.
- Feedback is formatted to reinforce notation and music theory rather than generic praise using research (Li (2025) and Xu (2025)).
- The output prioritises actionable steps to correct the specific "Worst Bar" identified by the algorithm.


```
{
  "bar_index": 1,
  "advice_text": "You played a D4 instead of E44. Focus on the correct notes in this passage.",
  "played_note": "D4",
  "expected_note": "E44"
}
```


It looks like you had a bit of a tricky spot with your finger placement during the transition in the first bar.

- In bar 1, there was a slight slip where an E4 was played instead of the F4 note which suggests the thumb tuck might need a bit more focus.
- You also played a G4 instead of the F4 in bar 1 while trying to keep up with the tempo of 120 beats per minute.
- Refining your hand shape as you move through bar 1 will ensure your pitch accuracy remains consistent throughout the scale.

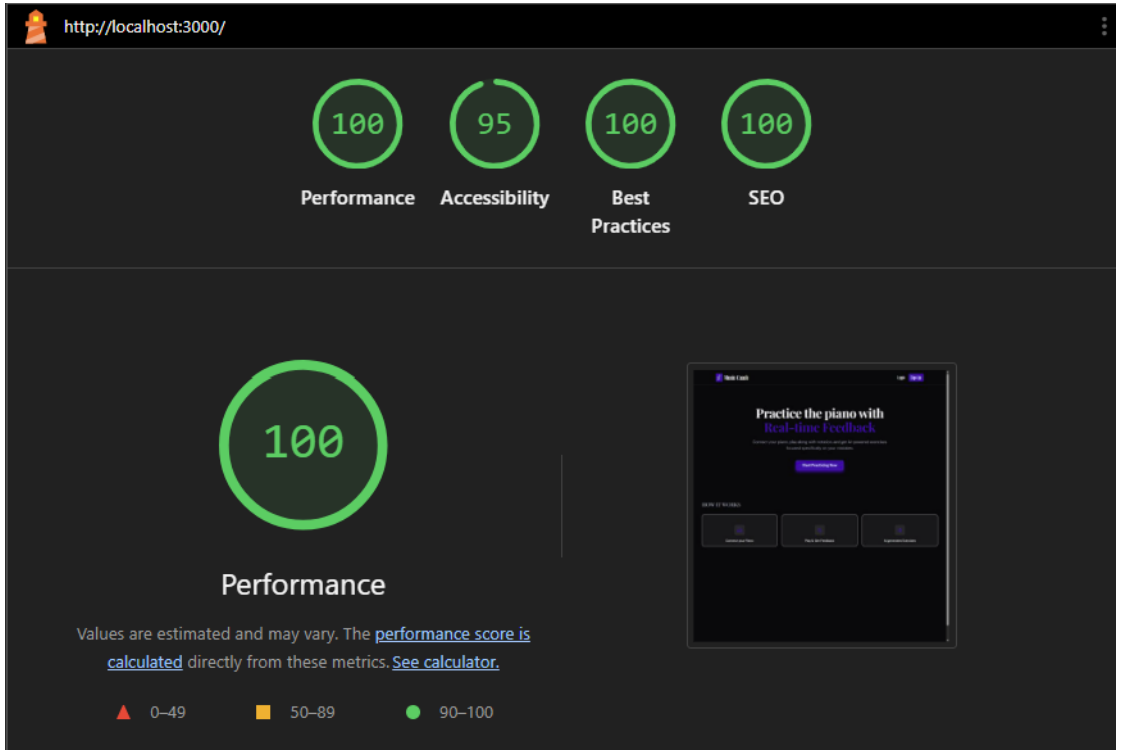
Testing - Quantitative



- Measured internal processing delay at 0.5ms, well within the 25ms target.

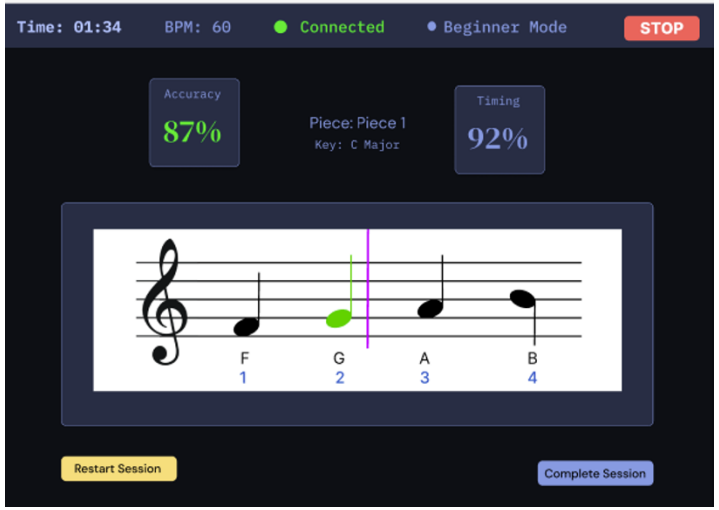
```
const handleNoteOn = useCallback((incomingNote: NoteMessageEvent) => {
  if (!isCapturingRef.current) return;

  //Latency testing
  const arrivalTime = performance.now();
  const latency = arrivalTime - incomingNote.timestamp;
  console.log(" Latency: " + latency.toFixed(2) + "ms");
  //
```

Lighthouse scores



- Achieved high Lighthouse score for performance and accessibility.

UI Review Testing – Qualitative #1



- Most participants were able to identify key features such as
  - the sign up and login buttons
  - the dashboard
  - MIDI connection status
  - the new session flow
- However, there were two key areas that required refinement
  - User uncertainty regarding advanced technical terms (e.g., "MIDI status", "BPM").
  - Feedback suggests the UI felt “too technical”
    - Target Audience Consideration

Functional Testing – Qualitative #2

User Perception & Critique

- Misleading title – “AI Music App”
  - Participants expected an application that generated music
- Lack of clarity in accessing past sessions indicating a clear drop in accessibility.
- Fixation on the visual music sheet caused users to completely bypass multiple UI features.

Design Updates

- Changed application title to Music Coach with accommodating subtitle.
- Redesigned Practice session to reduce cognitive load and improve simplicity.
- Redesigned dashboard design to user clear terms for navigation.

Summative Testing – Qualitative # 3

Metric	Average Score (Out of 5)
Ease of use and understanding	5.00
Straightforward setup process	4.86
Helpful real-time feedback	4.43
Clear post session summary insights	4.43
Likelihood of future use	4.43

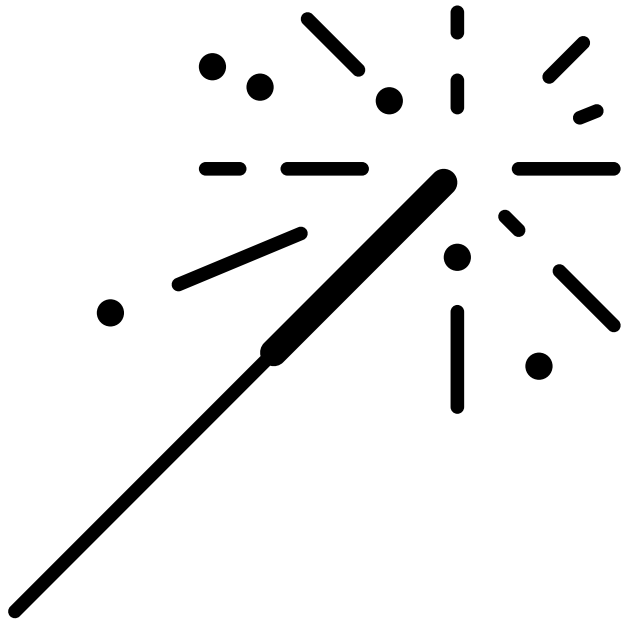
- 90% of testers successfully completed a remediation loop using the generated practice drills.
- Users identified “ease of use" as the primary advantage.
- Testers praised the interface for feeling like a professional tool rather than a game.
- Participants reported a higher confidence in self correction when provided with specific bar diagnostic

Limitations

- Monolithic Application
  - The application is currently tightly coupled. Future versions would benefit from a more modular, decoupled structure.
- Limited performance analysis
  - Analysis focuses on pitch and rhythm accuracy, with dynamics and articulation reserved for future development.
- Drills generated using snippets from original piece
  - Practice exercises are restricted to bars extracted from the original score rather than creating entirely new melodies
- The process of converting complex midi files into static reference files can occasionally lead to formatting errors

Future Enhancements

- Multi-instrument adaptions
  - Expanding the application to track other instruments beyond the piano.
- Upload MIDI files feature
  - Letting individual learners and teachers upload their own MIDI files to practice custom music.
- Implementing missed and skipped notes algorithms to enhance musical context
- Polyphony detection and analysing two hands
- Artificial Intelligence generates drills based on user's historical performance
  - Using past performance data to create custom practice drills trained on real performance files



Conclusion

- Music Coach bridges the pedagogical gap.
- Moves beyond gamification into a diagnostic application
- Successfully built a low-latency, hybrid architecture that bridges the gap between algorithmic note tracking and pedagogical coaching.

References

- Gomes, D. (2024). A Comprehensive Study of Advancements in Intelligent Tutoring Systems through Artificial Intelligent Education Platforms. In: *Advances in Educational Technologies and Instructional Design*. [online] IGI Global, pp.213–244. doi:https://doi.org/10.4018/979-8-3693-6170-2.ch008.
- Huang, N. and Ding, X. (2022). Piano Music Teaching under the Background of Artificial Intelligence. *Wireless Communications and Mobile Computing*, [online] 2022(1). doi:https://doi.org/10.1155/2022/5816453.
- Li, Y. (2025). The effect of real-time feedback generation algorithms on performance skill improvement in piano art instruction. *International Journal for Housing Science and Its Applications*, [online] 46(4). doi:https://doi.org/10.70517/ijhsa46413.
- Schedl, M., Gómez, E. and Urbano, J. (2014). Music Information Retrieval: Recent Developments and Applications. *Foundations and Trends® in Information Retrieval*, [online] 8(2-3), pp.127–261. doi:https://doi.org/10.1561/1500000042.
- Xu, X. and Zulkarnain, R. (2025). The impact of feedback-oriented teaching on piano learners' performance, satisfaction and motivation. *Advances in Curriculum Design&Education*, [online] 1(3), pp.1–10. doi:https://doi.org/10.63808/acde.v1i3.250.

THANK YOU FOR TAKING THE TIME TO READ MY PROJECT FINDINGS!